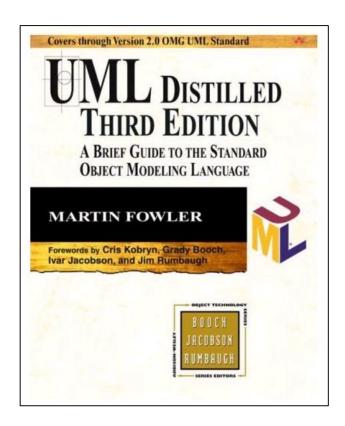


# **Unified Modeling Language (UML)**



Good tutorial on object-orientation development process and short (about 200 pages) introduction to the UML.

### **Unified Modeling Language**

- Graphical language for designing, documenting and communicating designs object-oriented software
- Standardized by the Object Management Group (OMG) and backed by a large group of software companies
- Allows the specification and description of object-oriented software at various levels of detail
- Supported by many different CASE tools for
  - Designing OO software
  - Documenting OO software
  - Reverse engineering OO software
  - Creating skeleton code for OO software
  - Refactoring OO software



# Overview of Basic UML Notation for Object Orientation (OO)

## Simple UML Class Diagram

#### Abstract class (italics) Abstract member function (italics) C BaseA aFunc(): int Dependency (relationship) Association (relationship) Abstract class Inheritance and inheritance (relationship) (italics) Role name BaseD **DerivedB** Concrete class (non-italics) Multiplicity **DerivedE** eFunc(in c: C) Member function implementation

(non-italics)

## Compatible C++ declarations

```
class BaseA {
public:
  virtual ~BaseA() {}
  virtual int aFunc() const = 0;
  virtual C\& c() = 0;
};
class DerivedB : public BaseA {
public:
private:
  std::vector<BaseD*> d ;
};
class C { ... };
class BaseD { ... };
class DerivedE : public BaseD {
public:
  void eFunc( const C& c );
};
```

